Corso
Lotus Domino Designer

Emanuele Tonelli
www.etonelli.com

1

Database o applicazioni?

- L'ambiente Notes permette di gestire informazioni destrutturate (documenti piuttosto che tabelle di record)
- Un database notes • una collezione di documenti omogenei
- Quando si definisce la struttura di un documento (campi presenti) si definisce anche il layout grafico

2

Database o applicazioni?

- Un database Notes corrisponde quindi di solito ad una applicazione
- Vi sono casi in cui un'applicazione • costituita da diversi database (uno principale e altri collegati anche in modo non visibile all'utente)
- Notes • molto lontano dalla logica dei relazionali (informazioni normalizzate e collegate con Join), qui le informazioni accessorie (tabelle collegate) vengono spesso importate nei documenti

3

Come si crea un'applicazione

- Creazione da modello (database Notes che contiene solo la struttura non i dati, formato ntf)
- Copia e modifica di un database esistente
- Creazione ex-novo (modello vuoto)
- Possibilità di copiare solo alcuni elementi di impostazione di un database esistente

4

Programmazione Notes

- Tramite linguaggio @formule, usate per le automazioni + semplici e immediate, funzionale (scarsamente procedurale)
  - Funzioni @: operano su date, numeri, stringhe, oggetti Notes (es. @year(Data) @name(...))
  - Comandi @: simula comportamenti dei menu
    - Es: @command([fileclosewindow])
- LotusScript: linguaggio procedurale con sintassi VB, ma utilizza classi con oggetti Notes (database, document, item)

5

Programmazione Notes (2)

- LotusScript: 2 tipi di classi
  - Classi Back-end: tutti gli oggetti Notes (database, view, document,...)
  - Classi Front-end: Interazione con l' utente (documento corrente, database corrente,...)
- Programmazione Java* (solo classi back-end)
- Programmazione Javascript* (a supporto della programmazione web)

* Non trattato qui

6

Programmazione a eventi

- E' possibile intercettare il verificarsi di determinati eventi sugli oggetti Notes e associare a questo del codice (programmi)
- Es queryopen di un document (quando un documento sta per essere aperto), querysave (tentativo di salvataggio, per controllo campi), regiondoubleclic in una vista
- Di solito • in LotusScript, ma può essere anche una formula

7

Elementi del design di Notes

- Form (modulo)
- View (vista)
- Page (pagina)
- Navigatore
- Outline (schema)
- Frameset
- Agenti

8

Form – Moduli

- Il form permette di definire la struttura dei documenti che si andranno a creare nel database.
- La struttura viene definita sia come tipologie dei dati che come layout
- Il form non corrisponde al documento, ma rappresenta la struttura con cui il documento viene creato, modificato o visualizzato
- Un documento può essere creato con un form e aperto con un altro

9

Form (2)

- Quando si crea un documento basato su un certo modulo nel documento viene creato un campo di nome Form con valore il nome del modulo stesso
- Quando si crea un documento basato un modulo la struttura del modulo non viene salvata nel documento (a meno che non sia stata impostata l' opzione"Registra modulo nel documento"); il documento se inviato in un altro database potrebbe non essere visibile o avere un layout molto diverso

10

Con quale form viene aperto un documento

- Solitamente un documento viene aperto con il form utilizzato per crearlo, ma ci sono altre possibilità:
  - Formula del form (proprietà della vista)
  - Form salvato nel documento
  - Campo form
  - Form di default del database

11

Form (3) – Proprietà più importanti

- Nome: come viene identificato il form
- Alias: segue il nome (separato con un pipe |)
  - Se usato diventa il vero nome del form, salvato nel campo dform (può essere rischioso)
- Tipo: documento, risposta, risposta a risposta
- Includi nel menu (può convenire disabilitarlo e creare un pulsante opportuno)
- Modulo standard: questo modulo viene usato per aprire documenti per cui non si trova un modulo corrispondente

12

Form (4) – Ancora sulle proprietà

- Registra modulo in documento: la struttura del modulo • registrata nel documento (occupa più spazio ed • meno flessibile, ma permette di visualizzare il documento anche in un altro db)
- Fondi conflitti di replica (limita i conflitti, non quando si modifica lo stesso campo)
- Formule ereditano valori... (utile per creare documenti collegati a quello corrente, es. le stampe, oppure response document)

13

Form (5): i campi

- Definiscono la struttura dei dati contenuti nei documenti
  - Campo modificabile: valore inserito dall' utente
  - Calcolato: valore inserito in automatico non modificabile in base a formule @, valore memorizzato nel documento
  - Calcolato in composizione: come precedente, ma non ricalcolato ad ogni apertura
  - Calcolato in visualizzazione: il valore calcolato non viene memorizzato nel documento (occupa meno spazio, non utilizzabile nelle viste)

14

Tipi di dati

- Testo (campo libero)
- Rich text (allegati, immagini, formattazione dei campi); non si visualizza il contenuto nelle viste
- Numero (possibilità di calcoli,...)
- Data ora
- Campi nome (nomi utente, prelevabili dall' address book)
- Password

15

Tipi di dati: campi parola chiave (a scelta multipla)

- Lista in finestra; pulsante di scelta (solo una scelta), casella di controllo; casella di riepilogo; casella combinata
  - Opzioni: aggiorna i campi cambiando la parola chiave, aggiorna le scelte aggiornando il documento (aggiornamento rapido senza forzare frefresh)
  - Scelta tra vaolri statici, viste o formula @dbcolumn("Notes":"Nocache"; "server01": "db1.nsf"; "viewarticoli"; 1)

16

Campi autore e lettore

- Campo lettore (restringe ACL)
  - Se un documento contiene un campo lettore solo gli utenti (o ruoli) contenuti in queste campo hanno accesso al documento
- Campo autore (estende ACL)
  - Gli utenti o i ruoli presenti in un campo autore che hanno accesso in redazione sul db (solo creazione) possono anche modificare il documento

17

Formattazione: tabelle, regioni di layout

- Tabelle: sono l'elemento principale di formattazione dei dati, conviene sempre allineare i dati con delle tabelle piuttosto che con tabulatori
- Nella versione 5 si può mostrare una riga per volta (tabelle con 'linguette' tabbed)
- Nelle regioni di layout i campi possono essere posizionati in modo preciso
- Non tutti i tipi di campo sono utilizzabili nelle regioni di layout (no rich text o liste in finestra)
- Sono spesso utilizzate per finestre di dialogo
  - (funzione @dialogbox o metodo DialogBox)

18

Note ID e Universal ID

- Note ID: identifica un documento all' interno diun database (8 caratteri esadecimali)
- Universal ID: identifica il documento in assoluto (32 caratteri esadecimali)
  - Funzione @DocumentUniqueID (si può usare nella vista con funzione @text)
  - In lotusscript GetDocumentByUNID metodo della classe notesdatabase
- $Ref contiene il l' UniversalIDdel padre

19

I Response document (documenti risposta)

- Quando si crea un modulo lo si può impostare come documento principale o come documento risposta (response document, o response to response)
- I RD servono per creare documenti in gerachia (collegati tra loro partendo da uno principale)
- Si possono far ereditare al RD alcuni valori del documento principale
- I response document contengono un campo speciale $ref che contiene l' UniversalIDdel padre

20

Le viste

- Permettono di visualizzare i documenti in formato tabellare (colonne) oppure in formato diario
- View selection: indica quali documenti vengono visualizzati nella vista
  - Alcuni esempi: Select @all, select Form= "Cliente", select Form = "Scheda" | @alldescendants
- Struttura possibile: standard o calendario
  - (in questo caso le prime colonna contiene un campo data ora e la seconda una durata, di solito sono nascoste)

21

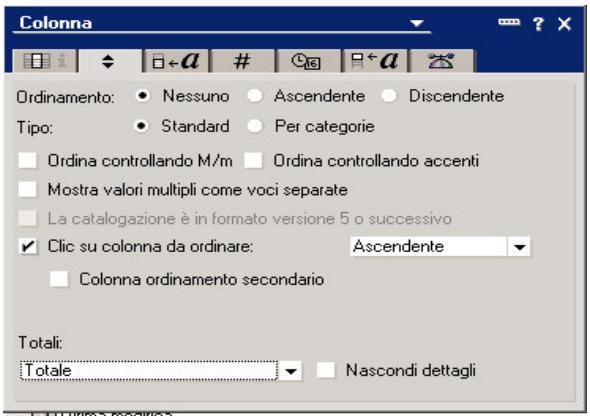
Le viste: proprietà principali

- Comprimi tutte alla prima apertura (per vista categorizzata): le categorie sono compresse
- Mostra doc risposta in gerarchia (permette di indentare i response documents, insieme alla prop della colonna Mostra solo risposte)

22

Colonne delle viste

- Nelle colonne si possono immettere funzioni semplici, campi o funzioni @
- Ordinamento e categorizzazione



Colonne delle viste (2)

- Icone nelle colonne: proprietà mostra valori come icone, la colonna deve contenere un valore numerico, sono già predefinite, non personalizzabili
  - (es. @if(@attachments>0;5;0): mostra una grafetta se ci sono allegati)
- Colonne nascoste: per permettere ordinamenti senza mostrare la colonna
- Funzioni usate nelle viste categorizzate
  - Funzione: @docchildren (figli di primo livello)
  - @docdescendants (tutti i discendenti)
    - Se combinate con un campo • necessaria

Viste: funzioni collegate

- DBCOLUMN (elenco) E DBLOOKUP (data una chiave torna un valore)
  - *@DbLookup("":"NoCache";"":"";"Funzioni";Codice;2)* – Restituisce il valore della colonna 2 della vista Funzioni dove nella colonna 1 il contenuto • uguale alla variabile Codice
  - Richiedono colonne ordinate, limite dei 64k di dati, si possono utilizzare anche in ambiente ODBC
- PICKLIST
  - *FIELD Scelta := @PickList((Custom):(Single); @DbName; "Prodotti"; "Selezionare un prodotto"; "Selezionare il prodotto da ordinare"; 2)* – Da utilizzare in operazione o pulsante: dalla vista Prodotti del db corrente si può selezionare un solo valore che va ne campo Scelta

Viste private

- Gli utenti possono (se viene dato accesso) crearsi viste private, le vedono solo loro
- Viste condivise, personali al primo uso: la struttura • definita dal designer, ma la prima volta che sono usate diventano private (spesso contengono una view selection basata sulla funzione @username)
- Le viste private possono essere memorizzate nel desktop dell' utente anzich nel database

Le cartelle

- Sono simili alle viste come struttura, ma non hanno una view selection, ma contengono i documenti che qui vengono spostati o copiati dagli utenti
- Possono generare confusione (cancellazione accidentale)
- Utilizzi: documenti preferiti, quando si effettua una ricerca i risultati possono essere messi in una cartella

27

Le operazione (actions) nelle viste e nei moduli

- Utilizzate per far comparire pulsanti nella barra sopra alla vista
- Semplici, formule o lotusscript (Java o Javascript)
- @command([compose]; "nomeform") per creare documenti, usato nelle viste
- Immagini da associare (dalla versione 5 anche personalizzabili)
- Possibilità di utilizzare operazioni condivise (la stessa operazione viene usata in vari punti dell' applicazione), migliore manutenzione

28

Schemi e navigatori

- Elementi che permettono la navigazione, di solito sono nella parte sinistra della finestra
  - Navigatori: statici, usati con le versioni precedenti, solo per compatibilità
  - Schemi: espandibili, flessibili (possono contenere sottomenu indentati), si possono inserire all' interno di pagine, per un miglior layout grafico
  - Conviene creare lo schema a dopo le vsite con la funzione genera schema predefinito
  - Notes 4-> Vari navigatori
  - Notes 5-> Pochi schemi (spesso 1 solo)

29

Pagine e frameset

- La pagina (page) • un elemento di design molto simile al form, solo che non può contenere campi
- Usata per inserire schemi incorporati o altri elementi incorporati, usata per il web
- Limite: per creare una pagina • necessario avere accesso in impostazione (design)
- I frameset permettono di gestire i frame anche in ambiente Notes (oltre che web)
- Nelle pagine come nelle form possiamo mettere HTML puro (html passante)

30

Inserimento di una vista in un modulo

- Una delle funzioni più interessanti di Notes 5 • la possibilità di incorporare una vista in un documento, questo viene fatto inserendo la vista nel form in fase di design
- Si può anche visualizzare solo una categoria di una vista categorizzata definendo la categoria in base ad una formula (es valore di un campo di un documento)
- Al massimo una vista per form, ma se ne possono avere altre in documenti associati
- Elemento di ' relazionalità'

31

La sicurezza

- Lista controllo accessi (ACL)
  - Composizione (solo aggiungere, no vedere), Lettura, Redazione (crea, ma non modifica), Revisione (crea e modifica), Impostazione (programma), Gestione (ACL)
  - Limite: il gestore può fare tutto
- Campi lettori e autori
- Cifrare i campi: rende inaccessibili i documenti anche in locale (sono cifrati con la chiave privata dell' ID dell' utente)
- Nascondere parti di form a determinati utenti (non • un vero elemento di sicurezza, i dati possono essere visti tramite le proprietà campi)
- Sezioni ad accesso controllato

32

I ruoli

- I ruoli permettono di creare accesso differenziato ad alcuni utenti senza inserire il nome dell' utente del codice, nel codice viene inserito il ruolo e poi gli utenti avranno o meno un accesso a seconda del fatto che hanno quel ruolo
  - Es. nascondere un parte di testo: si abilita la proprietà Nascondi paragrafo se • vera
    - @IsNotMember("[MANAGER]";@UserRoles) – Solo gli utenti con ruolo MANAGER hanno accesso al paragrafo
    - Nel codice i ruoli vanno scritti tra parentesi quadre

Gli agenti

- Sono programmi che operano su un insieme di documenti del database
- Lotusscript, formule o semplici
- Possono essere eseguiti dal client o dal server (in backuground pianificati)
- Gli agenti sul server non possono utilizzare oggetti di front end, ma solo classi di back end
  - Semplice agente che trasforma il campo Nome da "Carlo" a "Charlie"
  - FIELD Nome := @if(Nome="Carlo";"Charlie";Nome)
    - Può agire sui doc selezionati, su tutti i doc del database o su tutti i doc della vista

Elementi del design riutilizzabili

- Subform (sottomoduli): 'pezzetto' di form riutilizzabile
- Campo condiviso (le proprietà del campo sono definite una volta sola). NB: un campo presente in vari moduli con lo stesso nome non è detto che sia un campo condiviso
- Operazioni condivise
- Librerie di script

35

Formule maggiormente usate

- Formule maggiormente usate
  - @if(A>190;"Molto alto";A>180; "Alto";"Normale")
  - @AttachmentNames – Nome allegati
  - @char(65) A - @char(13) + @char(10)
  - @contains(@userroles; "[GUEST]") - @ismember(..)
  - @If(@IsNewDoc;"Nuovo documento";Subject)
  - @if(@iserror(@dblookup(...));"";@dblookup(..))
  - @implode("A":"B":"C") – "A B C"
  - @ReplaceSubString(Descrizione;@Newline;" ")
  - @subset($updatedby; -1) - @Subset(@DbName;-1)
  - @GetProfileField("Profilo interesse"; "Categorie profili")

36

@Formule (1) – Gestione dei dati

- @text(@year(data))
- @texttotime("01/03/2002")
- @Middle("North Carolina";"th";-2) - (= "or")
- Stringhe:
  - @trim(..), @left(..), @texttonumber(..), @length, @lowercase,@Propercase, @abstract(..)
- Data-Ora:
  - @adjust(..), @hour(..), @now, @today
- Nomi:
  - @name(@author;(CN)) – (Emanuele Tonelli)
- Sul documento:
  - @created; @accessed;@author

Funzioni Prompt e Picklist

- @Prompt((Ok);"Ricordare";"Non dimenticare di lanciare il backup stasera.")
- @Prompt((YesNo); "Invia memo?"; "Questo memo verrà inviato a chiunque sia elencato nei campi Per, CC e Car.")
- file := @Prompt((LOCALBROWSE) ; "Selezionare un Database da aprire" ; "1")
- Scelta := @PickList((Custom); ""; "Prodotti"; "Selezionare un prodotto"; "Selezionare il prodotto da ordinare"; 1)
- @PickList((Folders); (Single) server:database)
- FIELD Persona := Persona; @SetField("Persona"; @PickList((Name)))

Comandi maggiormente usati

- @command([fileclosewindow])
- @command([compose]; "documento")
- @command([EditInsertFileAttachment])
- @Command([ViewRefreshFields])
- @Command([SectionExpandAll])
- @Command([EditMakeDocLink])

39

Invio di un email con collegamento a un documento

```
tmpOggetto := Oggetto;
tmpAnno := Anno;
tmpNumero := Numero;
@Command([EditMakeDocLink]) ;
@Command([MailComposeMemo]);
@Command([EditGotoField]; "Subject");
@Command([EditInsertText]; "Determina " + tmpNumero + "\\" +
tmpAnno );
@Command([EditGotoField]; "Body");
@Command([EditInsertText]; "Oggetto");
@Command([EditInsertText]; @NewLine);
@Command([EditInsertText]; tmpOggetto);
@Command([EditInsertText]; @NewLine);
@Command([EditInsertText]; @NewLine);
@Command([EditInsertText]; "Fare clic per aprire la determina --> ");
@Command([EditPaste]);
@Command([EditNextField])
```

40

Lotus Script: classi di front end

- NotesUIWorkspace (area di lavoro di Notes)
  - Dim ws as new NotesUIworkspace
- NotesUIdatabase (db corrente)
  - Set uidb = ws.currentdatabase
  - Set db = uidb.database
- NotesUIdocument (documento corrente)
  - Set uidoc = ws.currentdocument
  - Set doc = uidoc.document
- NotesUIView
- ....

Classi LotusScript back-end

- NotesSession
  - Dim session as new NotesSession
- NotesDatabase
  - Dim db as notes database
  - Set db = session.Currentdatabase
- NotesView
  - Dim view as Notesview
  - Set view = db.getview("Vista1")
- NotesDocumentCollection
  - Dim col as NotesDocumentCollection
  - Set col = view.GetAllDocumentsbyKey(key, False)
- NotesDocument
  - Dim doc as NotesDocument
  - Set doc = col.getfirstdocument
  - While not (doc is nothing)
  - ---
  - Set doc = col.getnextdocument(doc)
  - Wend

Classi di gestione dei campi

- Classe NotesItem in NotesDocument
  - doc.Nominativo = "Marco" - equivale a
    - Set item = doc.appenitevalue("Nominativo", "Marco")
    - *Set item = Doc.ReplaceItemValue("V"&cstr(count),"1")* – Nome del campo costruito
  - If doc.nominativo (0) = "Marco" then ...
  - Assegnazione di valori multipli
    - Lettori(0) = "Marco"
    - Lettori(1) = "Matteo"
    - Set item= Doc.ReplaceItemValue("Lettori", lettori)
  - Aggiungere valori doc.AppendItelValue
- Classe NotesUIDocument
  - Call uidoc.FieldAppendText ("Body", "Leggere attentamente il messaggio")
  - composed = uidoc.FieldGetText("DateComposed")

Esempio codice LS (1)

```
Sub Queryopendocument(Source As Notesuiview, Continue As Variant)
        'non apro il documento associato ma il documento della segnalazione
        Dim docs As NotesDocumentCollection
        Dim doc As NotesDocument
        Set docs = Source.Documents
        Set doc = docs.GetFirstDocument()
        Continue = False

        Dim uidoc As notesuidocument
        Dim ws As New NotesUIWorkspace
        Dim session As New NotesSession
        Dim db As notesdatabase
        Dim view As notesview
        Dim segnDoc As notesdocument
        Set db = Session.currentdatabase
        Set view = db.getview("(Elenco)")
        If Not (view Is Nothing) Then
                Set segnDoc = view.GetDocumentbykey(doc.NumeroSegnalazione(0))

                Set uidoc = ws.editdocument(False, segnDoc)
        End If

End Sub
```

Esempio codice LS: ODBC

```
Option Public
Option Explicit
Uselsx "*LSXODBC"
```

```
Sub Initialize
        Dim ws As New notesUIWorkspace
        Dim uidoc As NotesUidocument
        Dim con As New ODBCConnection
        Dim qry As New ODBCQuery
        Dim res As New ODBCResultSet
        Dim db As notesdatabase
        Dim viewElenco As notesview
        Dim doc, newdoc As notesdocument
        Dim profiledoc As notesdocument
        Dim status As Integer
        Dim querystring As String
        Set db = ws.currentdatabase.database
        Set viewElenco =db.getView("(Elenco)")
        If Not con.ConnectTo("FonteODBC", "", "") Then
           Msgbox "Errore nella connessione alla fonte dati, verificare la configurazione del sistema"
           Exit Sub
        End If
        Set qry.connection = con
        querystring = "SELECT * FROM QLT WHERE NumScheda < 100 ORDER BY DataIns"
        qry.SQL = querystring
        Set res.Query = qry
        res.MaxRows = 0
        res.cachelimit = DB_NONE
        status = res.Execute ' si può testare status
        Do
                Set newdoc = db.createdocument
                newdoc.form= "Segnalazione"
                newdoc.numero= res.getvalue("NumScheda")
                ...
                'verifico se esiste già
                Set doc = viewElenco.getDocumentbykey(newdoc.numero1(0), True)
                If Not (doc Is Nothing) Then
                        doppio = True
                Else
                        newdoc.dataredazione = res.getvalue("DataIns")
                        newdoc.AziendaSegnalatrice = res.getvalue("SocIns")
                        ..
                        Call newdoc.save(True, True)
                        End If
                End If
```


Esempio (3): Export di un txt

```
Dim session As New notessession
Dim db As notesdatabase
Dim view As notesview
Dim doc As notesdocument
Dim sep As String
Dim riga As String
sep = ";"
Open "Elenco.txt" For Output As #1
Set db = session.currentdatabase
Set view = db.getview("Cartelle")
Set doc = view.getfirstdocument
While Not (doc Is Nothing)
        If doc.credito(0) > 0 Then
                riga = doc.CodiceFiscale(0) & sep & doc.Nominativo(0) & sep &
doc.indirizzofiscale(0) & sep & doc.cap(0) & sep &doc.comune(0) & sep &
doc.credito(0)
                Print #1, riga
        End If
        Set doc = view.getnextdocument(doc)
Wend
Close #1
```

Programmazione web: cenni

- Utilizzare form e viste diversi per il web e per Notes (dare lo stesso nome o lo stesso alias e rendere visibile il form e la vista solo per l'ambiente opportuno)
- Campo $$return – permette di personalizzare Form processed
  - @ReplaceSubstring("/"+@Subset(@DbName;-1) + "/RichiestaOK?OpenForm)";"\\";"/")
  - who:= @If(Left(From; " ") = ""; From; @Left(From; " ")); @Return("<h2>Grazie you, " + who + "</h2>
<h4><a href=/register.nsf/Main+View?OpenView>Main View")
- Applet ad hoc per Viste, rich text editor, schemi
- Prop: Usa Javascript generando le pagine (più pulsanti per pagina, più veloce, più funzioni supportate, no pulsante autom. submit)

47

Tips per lo sviluppatore

- File -> Strumenti -> Debug Lotusscript
- Viste nascoste: "(hiddenview)", utilizzate con tabelle di supporto (dbcolumn..) per vederle CTRL+SHIFT
- Indici delle viste (colonne ordinate o ordinabili) non abusarne troppo: occupano spazio e possono essere lunghe da creare (prima apertura)
- Evitare di dare la possibilità agli utenti di cancellare i documenti
- I pulsanti e le operazioni devono sempre tornare un valore
- Utilizzare i profiledocument

48

Tips per lo sviluppatore

- Il LotusScript utilizzare sempre Option explicit (dichiarazione obbligatoria delle variabili)
- % REM e % ENDREM per sezioni di commenti
- Aggiornamento di applicazione tramite Modello (grande flessibilità)
  - Modello: db vuoto ntf senza dati
  - DataBase -> Sostituisci impostazione
  - Nascondere il design: Abilitare Nascondi Formule e LotusScript in Sostituzione Impostazione
    - Quando si nasconde il design non si riesce più ad aprire (ma si può fare il refresh con altri modelli)

49

Tips per lo sviluppatore (3)

- Sviluppo client vs server
  - Su locale l'ACLnon viene considerata (a meno che non ci sia l'opzione Forza undLCA coerente, opzioni Avanzate in Lista Controllo Accessi)
  - E' possibile schedulare solo agenti sul server
- Problema del locking
  - Non esiste un meccanismo nativo di locking (problema della concorrenza, numerazione dei documenti)
    - E' necessario creare manualmente un meccanismo di numerazione con controllo della concorrenza

50